# BOOLEAN OPERATORS

***You're going to learn about boolean operators***

Comparison Operators → Logical Operators

OPTIONAL

for experienced programmers

# COMPARISON OPERATORS

**to be true or not to be true that's the question**

# EQUALITY OPERATORS

*These operators can be used with any comparable type and they yield a bool value*

| name | operator | example<br>*boolean expressions* | result |
|---|---|---|---|
| equal | == | 3 == 3 | **true** |
| not equal | != | "hello" != "hi" | **true** |

# ORDERING OPERATORS

*These operators can be used with any comparable type and they yield a bool value*

| name | operator | example<br>*boolean expressions* | result |
|---|---|---|---|
| less | < | 3 **<** 2.5 | **false** |
| less or equal | <= | 4.0 **<=** 4 | **true** |
| greater | > | "abc" **>** "cde" | **false** |
| greater or equal | >= | 'b' **>=** 'a' | **true** |

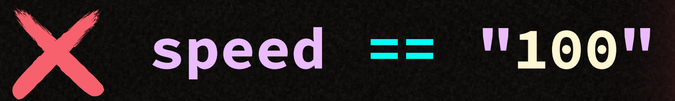
# CODING TIME!

# COMPARISON & ASSIGNABILITY

**when you can't assign then you can't compare as well**

# COMPARISON & ASSIGNABILITY

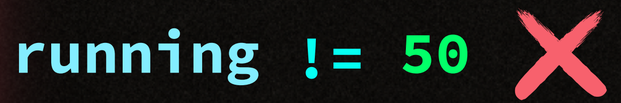
*when you **can't assign** then you **can't compare** as well*



```
func main() {
    var speed int
    speed = "100"

    var running bool
    running = 50

    var force float64
    speed = force
}
```



# COMPARISON & ASSIGNABILITY

*when you can't assign then you can't compare as well*

```
func main() {
    var speed int
    speed =

    var running bool
    running =

    var force float64
    speed =      force
}
```

# COMPARISON & ASSIGNABILITY

*when you **can't assign** then you **can't compare** as well*

✓ `speed >= 50`

`speed == int(force)` ✓

```
func main() {
    var speed int
    speed = 50

    var running bool
    running = true

    var force float64
    speed = int(force)
}
```

✓ `running == true` ✓

# CODING TIME!

# BOOLEAN OPERATORS

*Congrats!*

Comparison Operators ✓ → Logical Operators

# LOGICAL OPERATORS: AND

**“Logic will take you from A to B.Imagination will take you everywhere”**

***- Albert Einstein***

a logical operator

**combines**

bool expressions

and yields

a bool value

a logical operator
**can only work with**
bool expressions

shouldIStay := true

shouldIGo := false

shouldIStay **OR** shouldIGo

```
myAge := 25
yourAge := 20
```

~~myAge OR yourAge~~

# AND OPERATOR

*"Logical AND" operator can only be used with bool values*

| operand | operator | operand | result |
|---|---|---|---|
| true | && | true | true |
| true | && | false | false |
| false | && | true | false |
| false | && | false | false |

# CODING TIME!

# LOGICAL OPERATORS: OR

**“Right or Wrong, make a decision”**

***- Anonym***

# OR OPERATOR

*"Logical OR" operator can only be used with bool values*

| operand | operator | operand | result |
|---|---|---|---|
| true | \|\| | true | true |
| true | \|\| | false | true |
| false | \|\| | true | true |
| false | \|\| | false | false |

# CODING TIME!

# LOGICAL OPERATORS: NOT

# LOGICAL OPERATORS

| operator | expression | result |
| --- | --- | --- |
| && | *true* *true*<br>"hi" == "hi" && 3 > 2 | true |
| \|\| | *false* *true*<br>"hi" != "hi" \|\| 3 > 2 | true |
| ! | !("hi" != "hi" \|\| 3 > 2) | false |

# LOGICAL OPERATORS

## *NOT Operator Example*

```go
func main() {
    var on bool

    on = !on
    fmt.Println(on)

    on = !!on
    fmt.Println(on)
}
```

***NOT OPERATOR***

***!true** becomes **false***

***!false** becomes **true***

***!!true** becomes **true***

it can only be used with a **bool value**

```
$ go run main.go
true
true
```

# BOOLEAN OPERATORS

***Congrats!***

Comparison Operators ✓ → Logical Operators ✓

# summary

# comparison operators

equality operators

ordering operators

# equality operators

```
true == true   →   true
true != true   →   false
```

# ordering operators

```
1 < 2   →   true

2 > 5   →   false

2 >= 2  →   true

3 <= 2  →   false
```

# assignability and comparison

```
i, f := 15, 5.0
```

```
i > f
```

```
i > int(f)
```

a logical operator

**combines**

bool expressions

and yields

a bool value

# logical operators

**Logical AND Operator**

```
true && true      → true
```

**Logical OR Operator**

```
false || false    → false
```

**Logical NOT Operator**

```
!false            → true
!!false           → false
```

# short-circuit

```
false && true
```

runs

skips

```
true || false
```

runs

skips

# IF STATEMENT

*Congrats!*

Boolean Operators ✓ → If Statement → Error Handling → Simple Statement